Journal of International Technology and Information Management

Volume 23
Issue 3 *Double Issue 3/4*

Article 9

2014

Using Document Indexers for Faceted Search in Dataspaces

Hsun-Ming Lee
Texas State University

Jaymeen Shah
Texas State University

Ju Long
Texas State University

Using Document Indexers for Faceted Search in Dataspaces

Hsun-Ming Lee
Jaymeen Shah
Ju Long
Department of Computer Information Systems & Quantitative Methods
Texas State University
USA

ABSTRACT

Efficient information retrieval is essential to enrich user experience when searching for documents in dataspaces. With the continued growth in the volume and complexity of documents, the efficient information retrieval for searches has become increasingly challenging. To improve users' search experience, faceted search combines direct keyword search methods with faceted browsing using a predefined set of categories (facets). This paper studies a faceted search approach that integrates dynamic facets generation with search. To further enhance the faceted search, alternative indexers based on pre-defined ontology for data repositories within dataspaces are evaluated in terms of execution time and data size. Experimental results suggest that combining the proposed faceted search with appropriate indexers improves search performance enhancing user experience.

INTRODUCTION

Two main ways of information retrieval are browsing and searching. Browsing demands information seekers' attention and time, and when the tasks are repetitive, fatigue sets in quickly (Yeh & Liu, 2011). Browsing, especially through page after page of alphabetical listings with a browser, could be frustrating to users. Research has been done to study the approaches to make the browsing experience more efficient and effective. Grouping information as categories during browsing, also called faceted browsing or faceted navigation, is one of the approaches proposed for enhancing the users' browsing experience. Faceted browsing offers the users an overview of results, and narrows their list (Yeh & Liu, 2011). Since browsing naturally involves gathering data with conceptual grouping, faceted browsing has been demonstrated in facilitating more effective browsing and information retrieval, and in speeding up the users' decision making process (Kim et al., 2014).

Faceted search is a natural extension to faceted browsing. Searching is only effective if the query is formulated a priori (Kim et al., 2014). When the queries are difficult to define a priori, the search results may be irrelevant and the users may experience undesirable outcomes from their direct search alone. To make direct search more efficient, faceted search has been proposed. The faceted search approach integrates faceted browsing with direct search methods (Yeh & Liu, 2011), thus assisting users determine which portions of the facet contain more relevant and desired information (Perugini & Ramakrishnan, 2003). FLexible information Access using MEtadata in Novel COmbinations (FLAMENCO) was developed as an open source search interface framework (Yee et al., 2003). It utilizes faceted navigation for both browsing and searching. Since then, faceted searching has been used in various fields and has been proven to

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

generate better search results than direct search (Hearst, 2006). Faceted search solicits and captures search keywords, then prunes out branches of the hierarchy irrelevant to the user's information need (Perugini, 2010).

A major limitation of using faceted search is that categories are predefined and fixed regardless of the difference in search results. Recently, Kim et al. (2014) used semantic web to design a dynamic faceted search system. Their results showed that using it can save considerable time for users and improve the efficiency of search. Another key problem of a faceted search system is its limitation in searching within a very large and complex document domain that typically exists in big organizations and government agencies. These organizations usually create dataspaces to integrate and manage data and documents from many loosely connected heterogeneous data repositories. Some faceted-search systems show users all available facets and facet-values. This approach can quickly overwhelm the users and lead to diminished user experience and search performance (Sinha & Karger, 2005; Koren et al., 2008).

Based on the extant designs on faceted browsing, faceted search, and dynamic faceted search, our research considers the use of a complex ontology defined to support the dynamic category selection. In particular, the categories are generated for the navigation of a document with relations to subdocuments. The sample ontology (Figure 1) shows that a conference proceeding has relations to two groups of subdocuments: editors and articles. In addition, an article has relations to a group of authors. Eventually, all categories in faceted navigation may include all the properties of a proceeding and its subdocuments (such as title, year, publisher, article title, and article author's last name). A user enters keywords to form a query based on the ontology and receives a result including proceedings categorized and arranged by a faceted search system (Figure 2). The result may still include a large collection of proceedings. In this case, the user can issue an additional query before they want to start the navigation. In this proceeding example, we can apply keywords to match proceeding titles and article titles; therefore, each word in the titles can also be used as a category. For example, if the words Web and Database are used for the initial query, (title: Web) and (title: Database) can be considered as two categories. System's response in answering a keyword query can be quite time-consuming (Dong & Halevy, 2007) or ineffective (Liu et al., 2013), especially for documents with complicated structures within a dataspace. We propose to capture both text values and structural property using index structures. Our research aims to study system design for supporting robust queries of documents in the presence of a complex ontology, such that the system can efficiently provide dynamic faceted navigation combined with direct keyword search.

The remaining of the article is organized as follows. Next section reviews the related research to provide the necessary foundation for understanding faceted search systems. Then we provide an overview of the system architecture and evaluate the design of indexers used in the system. Finally, in the concluding sections of the paper we discuss the performance of indexers and benefits of using appropriate indexing scheme based on the ontology to enhance users' search experience.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Figure 1: Sample Ontology of Proceedings in a Bibliographic Database.

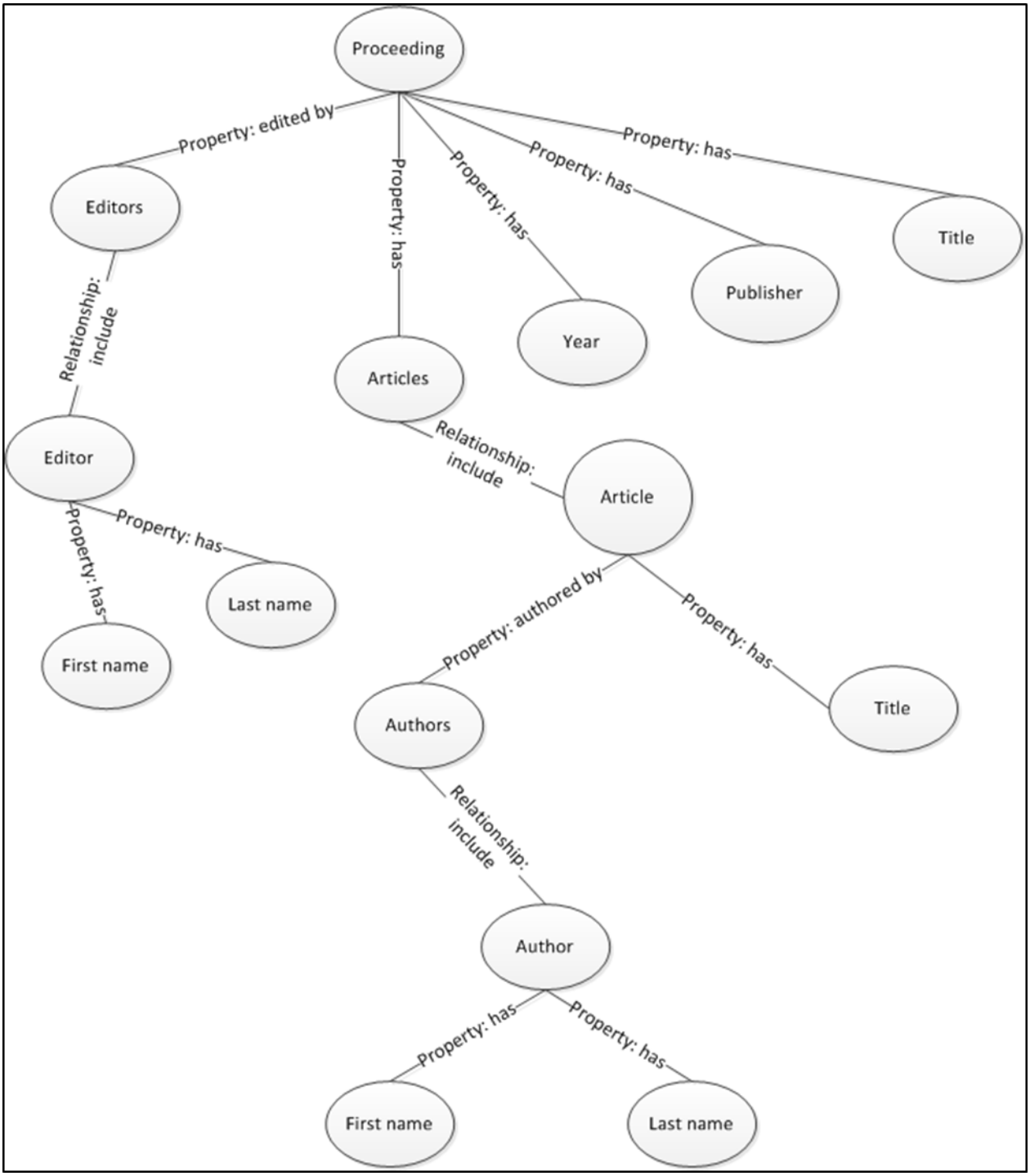


Figure 2: A User Interface Integrating Faceted Navigation and Direct Search on a Dataspace.

LITERATURE REVIEW

Faceted Browsing

Traditional browsing usually provides the user with an alphabetical listing of results and users often feel lost in the midst of pages of listings in the result (Yeh & Liu, 2011). To make browsing more effective, "good browseable interfaces should consist of rich scenes, full of potential objects of interest, which the eyes can take in at once, and then select items to give closer attention to" (Bates, 2007). Three functional requirements for effective browsing have been proposed: (1) enabling users to position themselves in an area of interest, (2) enabling users to recognize appropriate directions to further the search, and (3) enabling users to move quickly and efficiently throughout the system (Cox, 1992). Extant research has shown that faceted browsing can facilitate more effective and efficient information retrieval that meets the above criteria (Yeh & Liu, 2011). Facets are like different axes along which documents can be classified. Each document may be classified into one or more axes. In faceted browsing, each axis can be viewed as an entry point for users' search, and each item (document) may be accessed via many different entry points, therefore, the faceted browsing has inherent advantages over the alphabetical browsing (Garshol, 2004). This browsing approach addresses the proposed

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

rules for efficient browsing, such as providing as many potential objects of interest as possible for the eye to take in all at once and being able to view all entries, which enables users to position themselves in an area of interest (Bates, 2007; Cox, 1992). Faceted browsing offers several potential advantages to the user, including: more coherent and complete categories; more predictable categories; previews of where to go next; ways to return to previous states; logical alternatives; and fewer instances of empty results.

Faceted Searching

Searching is more effective when information need is known a priori so that more precise queries can be entered in the search box immediately at the beginning of the search process. However, often when direct search alone is used, users may get undesired results that are difficult to process and make sense because search results may include too many, too few, or irrelevant items. When faceted browsing is utilized alone, the user must browse through a list of category labels and may have to guess which one is most likely related to the topic of interest. This can be a time-consuming and frustrating process, especially, when there are many categories. Clearly, either faceted browsing or direct search by itself is generally not sufficient to achieve complex information search goals and may lead to an unsatisfying user experience or result sets (Olston & Chi, 2003). Therefore, research has proposed to combine these two approaches synergistically to enhance the search process and called it faceted search (Mackinlay & Zellweger, 1995). It is an alternative approach that combines faceted navigation together with the direct search. A representative faceted search interface is called "Flamenco", which provides the explicit exposure of hierarchical faceted metadata, both to guide the user toward possible choices and to organize the results of keyword searches (Perugini, 2010; Yeh & Liu, 2011). Apple's iTune store is another typical example of a faceted search interface that displays a relationship across several music attributes (or facets) such as song name, artist, and album (Perugini, 2010).

Dataspace

Dataspace is the abstraction that can be used to integrate and manage data from collection of loosely connected heterogeneous data repositories (Halevy et al., 2006). Most of the large organizations and government agencies have existed through the evolution in the approaches and data models used to store data. These organizations need to integrate enormous amount of interrelated data stored in a large number of diverse data repositories. However, they do not have a structured means to integrate interrelated data from the various loosely connected heterogeneous data repositories. Thus, developers have to repeatedly deal with low-level data management challenges across heterogeneous data repositories. Such data integration difficulties are commonly encountered in large organizations, government agencies, and libraries. In context of these organizations, usually there is a defined scope and control associated with the data integration needs which makes it possible to identify a data space that when properly managed and architected can provide significant benefits to the organization. In such scenarios, use of dataspaces can be leveraged to integrate data from loosely related diverse data repositories, such as relational databases, XML data repositories, text data files, and data repositories using other data models. This data integration approach differs from the classic data integration approach which requires considerable upfront effort for semantic integration across the data repositories before providing any services (Franklin et al., 2005). Further, in such organizational setting use

of relational databases alone is not adequate as large amount of loosely related heterogeneous data consisting of structured, semi-structured, and unstructured data have to be managed and integrated.

Dataspaces can be used to incrementally create and manage relationships among the various heterogeneous data repositories that exist within and across the organizational borders to provide required services to the users as needed. To enable users to meaningfully use the data stored in various existing large data repositories, search and integration of data from these data repositories is necessary. Use of faceted search within the dataspace can facilitate users' search process as they can use keywords for their search. In addition, when users' search is being performed within dataspaces associated with large volume of data, use of appropriate indexing is essential to support the need for fast response to users' search queries. Proper indexing of data and documents is critical to achieve fast response to users' queries in a highly heterogeneous and large data environment managed as a dataspace (Singh & Jain, 2011).

The dataspace schema mediates between the user and the various data repositories (data sources). One of the distinguishing characteristics of dataspaces is that the dataspace schema is created incrementally as needed to support users' search. Creation of dataspace schema requires semantic schema mapping with and across the various data source schemas to enable data integration across the various data repositories. This semantic schema mapping represents the relationships that exist between the various heterogeneous data repositories and the dataspace schema. As dataspace schema needs to dynamically evolve in order to provide relevant data in response to the users' search, the use of semantic ontology is studied and implemented in this research.

Semantic Web

Fixed category cannot dynamically evolve and provide relevant information by contextual grouping for the users when needed. Thus, the dynamic category selection using the Semantic Web technology has been proposed (Kim et al., 2014). For instance, categories such as song name, artist, and album in the iTune store interface aforementioned are all pre-defined and static. Selecting categories dynamically has several advantages since it could provide different categories for users according to their individual search contexts. If users query with different key words, they might assume different contexts and the natural categories provided with the results should be adapted to their situations (Kim et al. 2014). Research has shown that the dynamic faceted search offers users search experience that suit users' search contexts, and enable users to reach the items of interest much easily compared with the direct search. This saves users' time and improves the search efficiency.

The technology characterized as "Semantic Web" means that information is not only intended for human readers, but also for processing by machines, enabling intelligent information services, personalized Web-sites, and semantically empowered search-engines (Decker et al., 2000). To develop systems based on the Semantic Web, ontological models serve as building blocks for the tasks in representing the characteristics of data (Wang & Tanzel, 2013). In faceted search, the dynamic category generation compels selection of appropriate properties predefined in a given ontology (Zhu et al., 2013).

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Indexing Documents

Document indexing is a challenging task, which is classified as an NP-complete problem (Fraenkel, 1981). Although indexing is a difficult task, tremendous increase in the volume of documents that are being searched necessitates the use of document indexing. Search of the document base directly is a time consuming and processing intensive operation directly affected by the number of documents and size of each of these documents (Bouramoul, 2011). Thus, for a large volume of documents effective search and retrieval of documents is not possible without indexing. Creation of document indexes facilitate fast processing of user queries to search for documents of interest. The two types of indexing approaches include the traditional indexing that use a set of terms to describe the content of the documents and the semantic indexing that use an ontology. In both approaches documents are represented by a set of terms.

The traditional document indexing methods use inverted lists that are known to scale well. In this approach, a set of terms to document mapping and its inverse is used. It relies mainly on the occurrence frequency of each of the terms within the documents to be indexed. A term has a higher relevance if it occurs more often in documents than another term with fewer occurrences and which appears only in few documents within the document collection. This approach depends on the distribution of the terms within the document collection, thus works well when the search is performed on documents that mainly contain keywords instead of paragraphs with sentences (Hurst, 2003).

The traditional approach neglects the use of semantic relationships between the terms and its syntactic role within the documents. In semantic indexing, a domain ontology is used to derive a set of terms that describe the informational content of the documents. These terms typically corresponds to the node entries in the ontology (Bouramoul, 2011). The use of semantic relationship of terms within the documents is likely to improve the document retrieval performance in terms of relevance and speed (Tsatsaronis, Varlamis, & Norvag, 2012). Indexing method used to index documents must be easy to implement and efficient to store and retrieve documents. It must effectively control the increase in the index size, scale up as the document collection increases in size. In addition, semantic indexing should be capable of being applied across different domains.

ARCHITECTURE AND INDEXERS

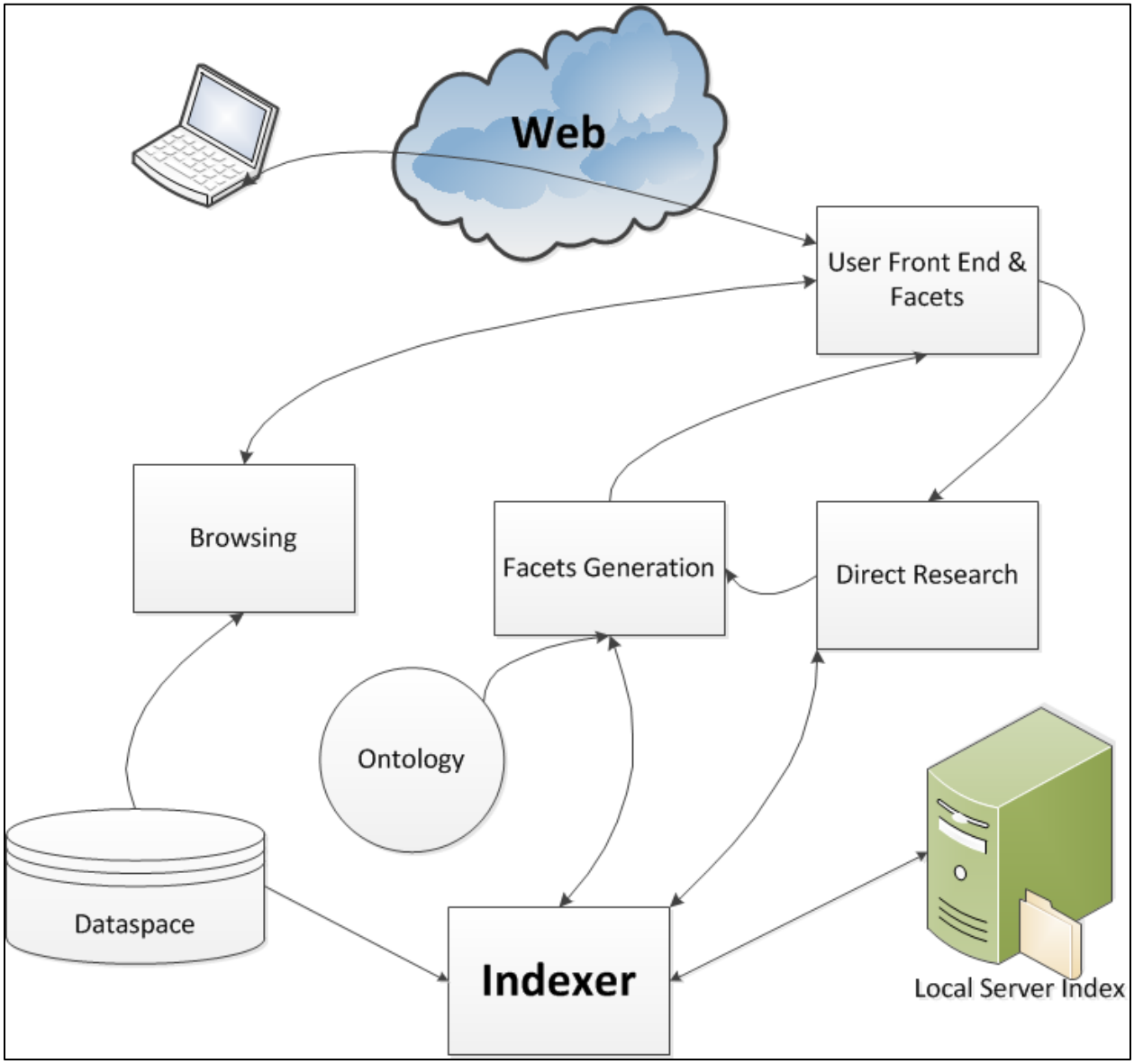
The integrated faceted search system has three major elements: data storage, indexing, and searching. Figure 3 depicts its general architecture. First, documents are loaded into data repositories. The ontology specifies the document structure and is used to generate browsing facets. Each document is indexed for fast retrieval. Users search for the indexed document directly using a keyword query and the system generates facets based on the documents' properties. Finally, the users browse the resulting set of documents following the facets presented on Web pages (see Figure 3). In this system, we use MongoDB to store and retrieve documents. The searching and indexing elements are implemented using the Apache Lucene, an open source library for text-based indexing and searching.

A key to ensure excellent system performance is a robust indexer, which analyzes the text-based

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

fields (properties) in the documents in order to enable fast queries. Using the fields as input to build a full-text index, the indexer must assure that it has capabilities to complete direct searches and support facet generation efficiently and correctly.

Figure 3: System Architecture for Faceted Search Using Ontology.



Indexes were generated using Lucene. An index consists of an inverted file; a word-oriented mechanism is used for indexing a collection of texts in order to accelerate the task of searching the documents (Armentano et al., 2014). Indexes are internally composed of a series of segments containing the following information:

- Term Dictionary is a dictionary containing all the terms used in all indexed fields of the entire collection of documents. The dictionary also stores the number of documents containing a term and pointers to the data frequency and proximity.
- Term Frequency for each word in the dictionary is designated to store the IDs of all documents that contain the term and frequency of the term in each document.
- Proximity Term for each term is the position in which the term occurs in each stored document.
- Term Vector is a vector of terms for each field in each document.

Lucene includes several analyzers, which examines the text values in fields and collects the terms that will be used for queries when indexing the document. The simplest analyzer splits the text into tokens using white spaces as separators. An indexer can be built using Lucene analyzers. In our system, a user applies direct search to refine the facets until targeted documents are identified. Assuming that users are non-experts and have no knowledge of the document structure, this type of search is keyword-based and returns any document containing the entered terms in the document's text fields. When a facet is selected, all subsequent searches will only show results matching the selected facet's property. Therefore, a facet selection can be implemented as another type of query with a predicate. For example, assume that the user searches for documents matching the keywords - ''Web Database John'' and a collection of 80 proceedings are returned. Then the user can select ''Web" under the Article Title category to narrow the search space. We need to develop two types of indexers to support this approach of faceted search.

One indexer is field-based, which analyzes each field individually in documents. Used for a predicate-based query, the created dictionary can indicate if a term exists in a document's field. Referring to the above example, the indexer can tell if a proceeding document contains an article including the term "Web" in the article title when the facet is selected.

The other indexer is path-based and designed for the direct (keyword) research. This indexer is necessary since it is difficult and sometimes even impossible, to formulate a search query that incorporates semantic knowledge in a clear and precise way (Cohen et al., 2003). The following example is used to demonstrate this issue. Suppose that a user is trying to find papers authored by Vianu on the topic of logical databases. This might be formulated as the search query: "Vianu logical databases." A site stores a proceeding containing two articles, where one article has title "Querying Logical Databases" and author "Moshe Y. Vardi" and another article has title "A Web Odyssey: From Codd to XML" and author "Victor Vianu." A standard search engine (such as field-based search) would regard the above document (proceeding) as an appropriate response, since it contains all the search terms. However, although the search terms appear, they do not all appear in the same context (article). Thus, the document is not an ideal response to the user's query. To address this problem arising because a complex document must be treated as an integral unit, many studies have adopted the "Path Return" technique to effectively identify desired results for keyword search (Bhalotia et al., 2002; Liu & Chen, 2007). In short, it considers a query answer as a document containing a directed path from the root to each keyword node. Thus, a path-based indexer is created with a meaningful purpose. Using the example in Figure 1, the path connected by a proceeding, an article, and an author provides a semantically appropriate index for a text-based keyword search.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Next, we will present the implementation of these indexers in Lucene and evaluate their performance.

EXPERIMENTAL RESULTS

Java programs were developed for this study, which were used to verify and validate the document indexers. To implement key parts of the proposed system, the program enables the following system functions: (a) to load XML-based data (documents) into a MongoDB database, (b) to build field-based and path-based indexers using proceeding titles, article titles, and author names, (c) to allow users to search the documents by matching entered keywords and predicates, and (d) to facilitate the construction of facets based on the ontology in Figure 1.

The experiments were performed on a 2.40GHz Intel Core i7 machine running Windows 7, with 8GB memory and one 349GB hard disk. The experiments were performed using the DBLP data set, which contains bibliographic information on major computer science journals and proceedings. Java programs developed for the study indexed 21,927 DBLP conference proceedings (1,269,330 articles totally) stored in MongoDB.

Using Lucene, we add each document (proceeding) to the analyzer and the document is indexed. We can customize fields used by the analyzer in several ways. First, a field can be stored but not analyzed, which means that terms used in the field are not separated and term positions are not stored by the indexer. Second, we can add many fields to the documents, but use of more fields degrades performance of the indexer. For this study, we built a field-based indexer based on three fields of a proceeding: the proceeding title, the title of each proceeding article, and the authors of each proceeding article. Since a proceeding contains many articles, each field such as the article title is added multiple times into the analyzer to build the field-based index for the document (a proceeding). To create a path-based index, we concatenate the article title and its author names for each article of a proceeding. This analyzer only uses a single field named "article path" and treats each article as a logical unit within the document. For each article, we also append its proceeding title to build a complete text for the "article path" field. Consequently, the path field is added multiple times for each proceeding (document).

A query is used to verify the indexers. First we retrieve the list of proceedings matching the key words "object database Wang" based on a path-based indexer. To be included in the search result, a document (proceeding) must have at least one article where its title, proceeding title, and author names contains these three words. Consequently 37 documents are returned, and these documents were checked visually to confirm that the result is correct. Next, we use the same key words to search for documents via a field-based indexer. In this case, a proceeding is matched if the three words exist in its title, a title of its article, or an author of its article. There are more than 5,000 documents included in the result of this search. This result is consistent with the observation made by Cohen et al. (2003).

To evaluate the performance of the two types of indexers, the time needed to index various numbers of documents is tested. The test (Table 1 and Figure 4) results show that the path-based indexer can be built in an efficient manner when the number of documents grows. However, it appears that the build time significantly increases for the field-based indexer when the number of

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

documents increases. The field-based indexer needs more computing resource since it uses multiple fields for indexing each document. It appears that the ratio between the build times of two indexers also grows as the document number increases. The query times for the path-based and field-based indexers increase linearly as the number of proceedings increase (Table 1 and Figure 5). The increase in query times for the field-based indexers is significantly faster than the query times for the path-based indexers.

Table 1: Test Results of Indexer Performance.

# of Proceedings	Total # of articles	Database Size (GB)	Build Time of a Path-Based Indexer (sec)	Build Time of a Field-Based Indexer (sec)	*Q1 (sec) Path-Based	**Q2 (sec) Field-Based
1000	56,144	0.203	7.20	109.60	0.031	0.094
2000	116,828	0.203	9.00	218.34	0.040	0.116
3000	168,459	0.453	13.58	305.78	0.031	0.110
4000	231,707	0.453	12.26	433.81	0.041	0.127
5000	283,392	0.453	17.25	612.79	0.040	0.130
6000	348,985	0.453	17.18	602.00	0.031	0.140
7000	394.654	0.953	20.98	698.85	0.040	0.125
8000	470,636	0.953	21.82	893.55	0.047	0.156
9000	506,877	0.953	22.82	864.90	0.047	0.156
10000	587,819	0.953	28.31	1962.15	0.047	0.156
11000	623,944	0.953	30.33	1610.00	0.047	0.156
12000	704,534	0.953	32.69	2223.78	0.047	0.156
13000	743,786	0.953	33.12	2537.30	0.047	0.172
14000	819,879	0.953	37.97	2650.33	0.047	0.172
15000	867,687	0.953	39.41	2821.24	0.054	0.187
16000	933,304	0.953	38.68	3698.807	0.047	0.171
17000	988,485	1.953	41.48	3213.30	0.046	0.171
18000	1,047,767	1.953	45.61	2550.19	0.055	0.184
19000	1,104,150	1.953	49.17	3439.67	0.062	0.188
20000	1,161,378	1.953	54.11	3480.97	0.050	0.198

*Query matching the key words "object database Wang" based on a **path**-based indexer.
Query matching the key words "object database Wang" based on a **field-based indexer (max hit: 5000).

Figure 4: Number of Proceedings and Build Times for the Two Indexing Methods.

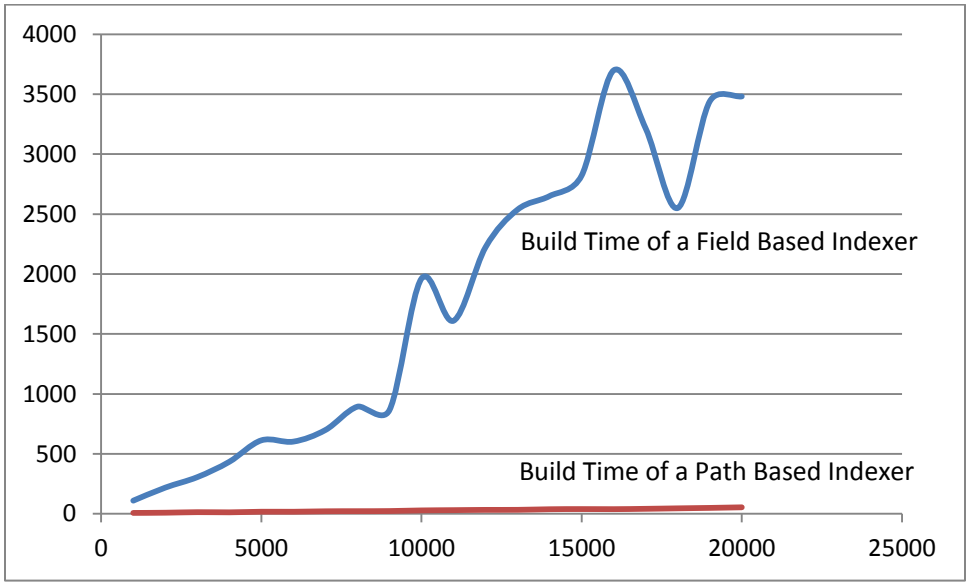
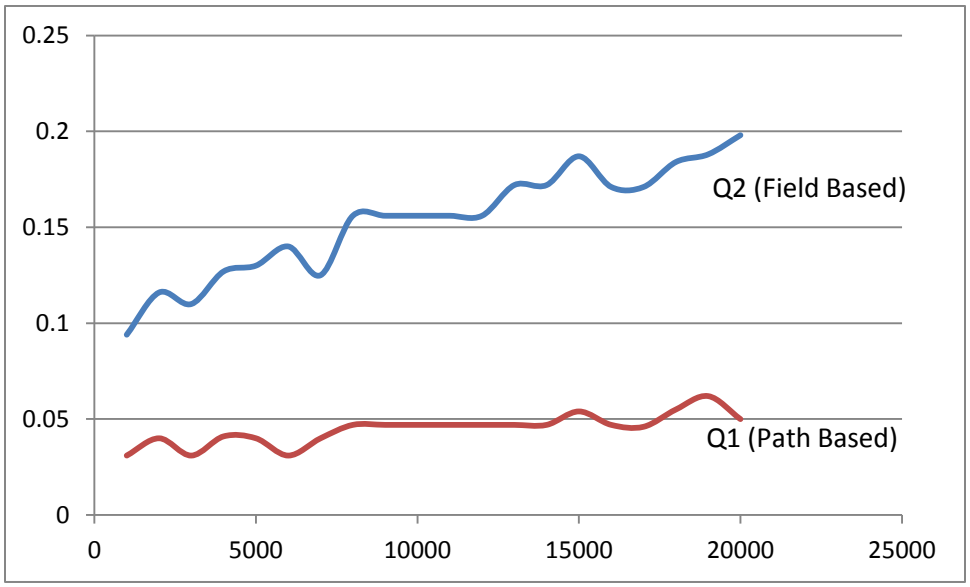


Figure 5: Number of Proceedings and Query Times for the Two Indexing Methods.



CONCLUSIONS

Efficient information retrieval is essential to enrich user experience with the process of searching documents in dataspaces. A document search system capable of providing an effective method to search documents relevant to users' search is vital as the volume and complexity of the document collection increases. In this research a faceted search that integrates direct (keyword) search, faceted browsing, and dynamic facets generation is studied as a solution to produce effective search results within dataspaces. In addition, document indexing based on pre-defined ontology for data repositories within dataspaces is critical to improve the efficiency of faceted

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

search. These capabilities are likely to enhance users' satisfaction with the results generated when searching through large volume of documents with complex semantic structure.

With the properties predefined in a complex ontology, this study evaluated design alternatives that are suitable for indexing documents to support the faceted document search system. The design treated the entire document as an integral unit. Using the open source text search engine - Apache Lucene, ontology properties were used to build indexes for faceted browsing of a collection of documents. To build path-based indexes for keyword search, input paths linked by multiple ontology properties were added into the text search engine. On the other hand, the field-based indexes were created using ontology properties individually.

Experiments conducted for this study indicated that the system based on the proposed architecture can be developed and implemented using open-source software. Performance of the two indexing strategies was tested using the DBLP data set stored in a MongoDB database. Alternative indexers were evaluated in terms of the execution time and size of the data. Using Lucene, the system can build indexes for keyword search. A field-based index can be directly created without knowing the ontology structure of searched documents in the dataspace. On the other hand, system developer must analyze the document ontology to create a path-based index. A major difference between the two indexing approaches is their ability to deal with the complexity of the documents to be indexed. Time required to build the index is another major difference between these indexing approaches. The system team must invest additional time to analyze the document ontology for designing a path-based index. In return, the computer uses much less time to physically create the path-based index and the users' search will be potentially more efficient. Experimental results suggest that it is necessary to carefully select the indexers to ensure effective search performance. Combining the faceted search with appropriate indexers will improve users' experience with the search process when searching for complex documents within dataspaces.

REFERENCES

Armentano, M. G., Godoy, D., Campo, M., & Amandi, A. (2014). NLP-based faceted search: Experience in the development of a science and technology search engine. *Expert Systems with Applications*, *41*(6), 2886-2896.

Bates, M. J. (2007). What is browsing—really? A model drawing from behavioural science research. *Information Research*, *12*(4), paper 330. Retrieved from http://InformationR.net/ir/12-4/paper330.html

Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., & Sudarshan, S. (2002). Keyword searching and browsing in databases using BANKS. In *Proceedings of the 18th International Conference on Data Engineering*, 431-440.

Bouramoul, A. (2011). The semantic dimension in information retrieval, from document indexing to query reformulation. *Knowledge Organization*, *38*(5), 425-437.

Cohen, S., Mamou, J., Kanza, Y., & Sagiv, Y. (2003). XSEarch: A semantic search engine for XML. In *Proceedings of the 29th VLDB conference*, 45-56.

Cox, K. (1992). Information retrieval by browsing. In *Proceedings of the 5th International Conference on New Information Technology*, Hong Kong, 69–80.

Decker, S., Melnik, S., Van Harmelen, F., Fensel, D., Klein, M., Broekstra, J.,Erdmann, M. & Horrocks, I. (2000). The semantic web: The roles of XML and RDF. *IEEE Internet Computing*, *4*(5), 63-73.

Dong, X., & Halevy, A. (2007, June). Indexing dataspaces. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 43-54.

Fraenkel, A. S. (1981). Document classification, indexing and abstracting may be inherently difficult problems. *ACM SIGIR Forum*, *16*(1), 77-82.

Franklin, M., Halevy, A., & Maier, D. (2005). From databases to dataspaces: A new abstraction for information management. *ACM Sigmod Record*, 34(4), 27-33.

Garshol, L. M. (2004). Metadata? Thesauri? taxonomies? topic maps! Making sense of it all. Retrieved from http://www.ontopia.net/topicmaps/materials/tm-vs-thesauri.html

Halevy, A., Franklin, M., & Maier, D. (2006, June). Principles of dataspace systems. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 1-9.

Hearst, M. (2006). Design recommendations for hierarchical faceted search interfaces. In *ACM SIGIR workshop on faceted search*, 1-5.

Hurst, W. (2003). Indexing, searching, and skimming of multimedia documents containing recorded lectures and live presentation. In *Proceedings of the eleventh ACM international conference on Multimedia*, 450-451.

Kim, H. J., Zhu, Y., Kim, W., & Sun, T. (2014). The dynamic faceted navigation in decision making using the semantic web technology. *Decision Support Systems*, *61*, 59-68.

Koren, J., Zhang, Y., & Liu, X. (2008). Personalized interactive faceted search. In *Proceedings of the 17th international conference on World Wide Web*, 477-485.

Liu, X., Chen, L., Wan, C., Liu, D., & Xiong, N. N. (2013). Exploiting structures in keyword queries for effective XML search. *Information Sciences*, *240*, 56-71.

Liu, Z., & Chen, Y. (2007). Identifying meaningful return information for XML keyword search. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 329-340.

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

MacKinlay, J. D., & Zellweger, P. T. (1995). Browsing vs. search: can we find a synergy? (panel session). In J. Miller (Chair), *Conference companion on human factors in computing systems*, 179–180. New York, NY: ACM.

Olston, C., & Chi, E. H. (2003). ScentTrails: Integrating browsing and searching on the web. *ACM Transactions on Computer-Human Interaction*, *10*, 177–197.

Perugini, S. (2010). Supporting multiple paths to objects in information hierarchies: Faceted classification, faceted search, and symbolic links. *Information Processing & Management*, *46*(1), 22-43.

Perugini, S., & Ramakrishnan, N. (2003). Personalizing web sites with mixed-initiative interaction. *IEEE IT Professional*, *5*(2), 9–15.

Singh, M., & Jain, S. K. (2011). A survey on dataspace. In *Advances in Network Security and Applications*, 608-621, Springer Berlin Heidelberg.

Sinha, V., & Karger, D. R. (2005). Magnet: Supporting navigation in semi-structured data environments. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, 97-106.

Tsatsaronis, G., Varlamis, I., & Norvag, K. (2012). SemaFor: Semantic document indexing using semantic forests. In *Proceedings of the 21st ACM international conference on Information and Knowledge Management*, 1692-1696.

Wang, H. T., & Tanzel, A. U. (2013). Composite Ontology-Based Medical Diagnosis Decision Support System Framework. *Communications of the IIMA*, *13*(2), 43-52.

Yee, K.-P., Swearingen, K., Li, K., & Hearst, M. (2003). Faceted metadata for image search and browsing. In G. Cockton & P. Korhonen (Eds.), *Proceedings of the ACM international conference on human factors in computing systems (CHI)*, 401–408. New York, NY: ACM Press.

Yeh, S. T., & Liu, Y. (2011). Integrated Faceted Browser and Direct Search to Enhance Information Retrieval in Text-Based Digital Libraries. *International Journal of Human–Computer Interaction*, *27*(4), 364-382.

Zhu, Y., Jeon, D., Kim, W., Hong, J. S., Lee, M., Wen, Z., & Cai, Y. (2013). The Dynamic Generation of Refining Categories in Ontology-Based Search. *Semantic Technology*, 146-158, Springer Berlin Heidelberg.

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

This Page Left Intentionally Blank

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy